Inductive Inference Without Overgeneralization From Positive Examples

Shyam Kapur*
Gianfranco Bilardi*

TR 89-1057
November 1989

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*Supported in part by the National Science Foundation under grant MIP-86-02256.

INDUCTIVE INFERENCE WITHOUT OVERGENERALIZATION FROM POSITIVE EXAMPLES

Shyam Kapur[1] Gianfranco Bilardi[1]

Abstract. Language learnability is investigated in the Gold paradigm of inductive inference from positive data. Angluin gave a characterization of learnable families in this framework. Here, learnability is studied when the learner obeys certain constraints. These constraints have been suggested by some studies of child language acquisition. Learnable families are characterized for learners with the following types of constraints: (a) conservative, consistent, and responsive, (b) conservative and responsive, (c) conservative and consistent, and (d) conservative. It is shown that the class of learnable families strictly increases going from (a) to (b) and from (b) to (c), while it stays the same going from (c) to (d).

1 Introduction

In his landmark paper (1967), Gold presented a paradigm of inductive inference, *identification in the limit*, which has been extensively investigated and developed. (For example, see Osherson, Stob & Weinstein (1986), and Angluin & Smith (1983).) Informally, in this model the *inductive inference machine (IIM)* is presented with a language L that belongs to a given family of languages F. The presentation consists of an infinite sequence of strings $s_1, s_2, \ldots$ where each of the strings of L appears at least once. (This is *positive* presentation. In a *complete* presentation where both positive and negative data are made available, every string from Σ^* is presented marked appropriately for L.) With each new string the learner is given the opportunity to "guess" a language from F. The IIM is considered a successful learner of L if, from some point onwards, all of its guesses coincide with L. The IIM learns the family F if it learns each of its languages successfully.

While in the situation in which both positive and negative data are available to the learner, learnability of the lower three levels of the Chomsky hierarchy follows, if only positive data is available, Gold showed that even a simple family containing every finite language and any single infinite language cannot be inferred. Angluin (1980) compared the two positions with regard to the availability of learning data.

[1]Department of Computer Science, Cornell University, Ithaca, New York 14853. Supported in part by the National Science Foundation under grant MIP-86-02256.

> Intuitively, an added difficulty in trying to do inference from positive rather than positive and negative data is the problem of "overgeneralization". If in the course of making guesses the inference process makes a guess that is overly general, i.e. specifies a language that is a proper superset of the true answer, then with positive and negative data there will eventually be a counterexample to the guess, i.e. a string that is contained in the guessed language but is not a member of the true language. No such conflict with examples will occur in the case of inference from positive data.

This restrictive nature of learning from positive data may actually enhance interest in it, because, for example, if children actually learn from only positive evidence (studies of language acquisition in children suggest that), it is even more likely that learning theory could suggest significant constraints on the structure of natural languages.

Angluin (1980) gave a complete characterization of the families learnable from positive data. She showed that the essential condition is the existence of a procedure that, for each language in the family, enumerates a finite set of which that language is a least upper bound (l.u.b.) with respect to the family. In this context the learner is allowed to make use of *indirect negative evidence* (as discussed in Berwick (1985)), that is, to change a guess consistent with the currently available evidence on the basis that a certain data element has not been observed within a certain interval of time.

Angluin (1980) also initiated the study of *conservative* learners. A conservative learner is one that does not change its guess unless evidence inconsistent with it is encountered. It can easily be shown that a family can be learned by a conservative learner if and only if it can be learned by a non overgeneralizing learner, that is, one that never guesses a subset of a previous guess. Angluin (1980) established that conservativeness actually restricts the class of learnable families, and derived a sufficient (but not necessary) condition for conservative learning.

In a framework similar to that in Angluin (1980), (more formally defined in Section 2), we investigate learnability under the conservativeness constraint and obtain a characterization for the learnable families. A variety of constraints on learners, motivated by studies of natural learning situations, have been proposed in literature (Osherson, Stob & Weinstein, 1986). *Consistency* (at any stage of the learning process, the guessed language must contain all the evidence), and *responsiveness* (a guess must be made for any new piece of evidence) are two such constraints. We will investigate the impact of both these constraints on conservative learning.

In Section 3, we begin the study of responsive and consistent conservative learning. It is easy to show that, in response to an input set of strings S, a conservative learner must always guess a least upper bound (l.u.b.) for S in F. This has been referred to as the *subset principle*[2] by Berwick (1985). We introduce the notion of an *f-algorithm*, a procedure that, at the ith stage, leaves its guess unchanged if it is consistent with the newly seen string s_i and otherwise changes it to $f(\{s_1, s_2, ..., s_i\})$, a new language depending only on the input strings seen up to that stage (not on the order or the number of occurrences of the strings), and always an l.u.b. of the set. An f-algorithm is not necessarily a (correct) learning algorithm, even if applied to a family that is conservatively learnable. However, we show that the existence of a procedure to obtain an l.u.b. language of the available evidence, which associates each language to at least one of its subsets, characterizes the families learnable in a (a) conservative, consistent and responsive way.

In Section 4, we consider how the above learnability conditions can be simplified for special families of languages such as those that admit an effective topological sorting, and those that are linearly ordered. In Section 5 we demonstrate a family that is not learnable by case (a) learner, but is learnable if we drop the consistency constraint. We also show there that a nonresponsive learner learns the same families whether or not it is consistent, and that a nonresponsive learner learns a strictly larger class of families than a responsive, conservative (possibly inconsistent) learner. In Sections 6 and 7 we characterize the learnable families for a learner which is: (b) conservative and responsive, (c) conservative and consistent, and (d) conservative.

Unlike in case (a), the learner we use for characterizing cases (b) and (c) update their guesses not only as a function of the set of strings observed, but also of the stage at which the guess is made. Curiously, the dependence upon the latter parameter can be dropped if the family being learned contains no

[2]Berwick (1985) argued that this principle can account for several linguistic phenomena. Since then, the subset principle has been the subject of extensive discussion in linguistics (Chomsky (1986), Manzini & Wexler (1987), Lust (1987), Williams (1987), and Kapur, Lust, Harbert & Martohardjono (1989)). However, its formal basis and significance have often been misunderstood. (Already in Berwick (1985) the principle is erroneously assumed equivalent to a condition due to Angluin (1980), which is reviewed below as Theorem 1 in Section 2.)

finite languages. It is also interesting to observe that, while Angluin's characterization for learnable families is based on being able to obtain for each language a finite set of which that language is an l.u.b., our characterizations are based on being able to obtain, for finite sets in a suitable class (different in cases (a), (b), and (c)), an l.u.b. language of that set. For conservative learning, case (d), we also obtain a characterization in the spirit of Angluin's result. Essentially, we show that conservativeness corresponds to computability, rather than recursive enumerability, of a finite set of which a given language is an l.u.b..

2 Background

Let Σ be a finite alphabet and Σ^* be the set of all finite strings formed by concatenating elements of Σ. A *language* L is a subset of Σ^*. A family of non-empty languages $F = \{L_1, L_2, \ldots\}$ is called an *indexed family* if there is an algorithm to decide whether x belongs to L_i for every x in Σ^* and every positive i. Clearly, the languages of an indexed family are *recursive*. (Soare (1987, Definition II 2.5) calls this a *uniformly recursive sequence* of languages.)

A *guessing algorithm* is a procedure whose input is an infinite sequence of strings $s_1, s_2, \ldots$ of Σ^* and whose output is a sequence of nonnegative integers $g_1, g_2, \ldots$. The procedure works in stages. At the ith stage, s_i is input and g_i is output. The intended interpretation is as follows: At the ith stage, if $g_i = 0$, then the algorithm makes no guess; otherwise, the algorithm guesses the language L_{g_i}. (We are thus assuming the algorithm is *prudent* (Osherson, Stob & Weinstein, 1986).)

A guessing algorithm is a *learning algorithm* for the indexed family F if, for any input sequence $s_1, s_2, \ldots$ such that $L = \{s_1, s_2, \ldots\} \in F$, there is a k such that $L_{g_k} = L$ and, for $i > k$, $g_i \in \{g_k, 0\}$. Intuitively, from some point onward, the value of the guess stabilizes to an index for the input language. (For conservative learning this is equivalent to EX-identification. See Angluin & Smith (1983).)

A guessing algorithm is *consistent* if it never guesses a language that does not contain all the data on which the guess is based. Formally, if $g_i \neq 0$, then $\{s_1, s_2, \ldots, s_i\} \subseteq L_{g_i}$. A guessing algorithm is *conservative* if it does not change its output unless it is inconsistent with the data. Formally, if $g_i \neq 0$ and $\{s_1, s_2, \ldots, s_{i+1}\} \subseteq L_{g_i}$, then $g_{i+1} = g_i$. A guessing algorithm is *responsive* if it produces a guess after every input string. Formally, for all $i > 0$, $g_i \neq 0$.

A central role is played by the relationship that holds between a finite set T and a language L in a family F when T is a subset of L but T is not a subset of any language in F properly contained in L. This relation will be expressed by saying either that T is a *tell-tale* subset of L, or that L is a *least upper bound* of T. As we shall see, various forms of computable mappings from (indices of) languages to tell-tale subsets and from finite sets to (indices of) least upper bound languages are intimately related to various forms of learning. The importance of tell-tale subsets was first indicated by the following result.

Theorem 1. (Angluin (1980)) There is a learning algorithm for an indexed family F if and only if there is an effective procedure that, on input i, recursively enumerates a tell-tale subset T_i of L_i. (The procedure need not signal or halt when the entire set has been output.)

Angluin (1980) established that the conservative constraint actually restricts the class of learnable families, and derived a sufficient condition for responsive and consistent, conservative learning. Our goal in the following sections is to obtain necessary and/or sufficient conditions for conservative learning in various combinations with other constraints.

3 Responsive and Consistent Conservative Learning

We begin with some crucial definitions. Let f be a positive integer function defined for all non-empty finite subsets of the languages in the indexed family F. We say that f is a *least upper bound* (l.u.b.) function for F if for each S, $L_{f(S)}$ is a least upper bound of S. An l.u.b. function f is *full-range* if each L in F has a subset S such that $L_{f(S)} = L$.

Definition 1. Given a computable l.u.b. function f for an indexed family F, an *f-algorithm* is a guessing algorithm that, at the ith stage, maintains its guess if consistent with s_i, and changes it to $f(\{s_1, s_2, \ldots, s_i\})$ otherwise. □

Notice that an f-algorithm is responsive, consistent, and conservative (RCC). Considering the simplicity of f-algorithms, it is of interest to investigate their learning abilities. It is easy to see that, at any stage, the language guessed by an RCC learner must be an l.u.b. of the input seen up to that stage. It is natural to wonder whether any RCC learner is an f-algorithm for some choice of f. The answer is negative since an RCC learner can produce different guesses for the same set of input strings when the set is produced in different orders.

Proposition 1. There are RCC learners which are not f-algorithms.

Proof. Consider the family $F = \{L_1, L_2, L_3\}$ where $L_1 = \{a,b\}$, $L_2 = \{a,b,c,d\}$, and $L_3 = \{a,b,c,e\}$. It is easy to construct an RCC learner that, for the input sequence $a,b,c,\ldots$ outputs the guess sequence $1,1,2,\ldots$, while for the input sequence $b,a,c,\ldots$ outputs the guess sequence $1,1,3,\ldots$. So, g_3 is different in the two cases, whereas it would be the same for any f-algorithm, since in both cases $g_3 = f(\{a,b,c\})$. □

Another natural question is whether any f-algorithm for an RCC-learnable family is indeed a learning algorithm. Even assuming that f is full-range, (otherwise some languages will never be guessed by the f-algorithm), the answer is negative.

Proposition 2. There is an RCC learnable family F and a full-range l.u.b. function f for F whose corresponding f-algorithm is not a learner.

Proof. Consider the family F: for all $k \geq 2$, $L_k = \{0\} \cup \{1,2,\ldots,k-1\}$, and $L_1 = \{1,2,\ldots\}$. The f-algorithm corresponding to the full-range l.u.b. function f such that $f(\{0\}) = 2$, $f(\{1\}) = 1$, and for all other S, $f(S) = \max(S)+1$ is not a learner for F. RCC learnability of F follows from Theorem 2 below. □

We now introduce a particular type of l.u.b. function f whose corresponding f-algorithm is always a learner.

Definition 2. Let f be an l.u.b. function for the family F. Let $L \in F$, and T be a finite subset of L. We say that T is *reserved* for L if, for every finite $S \supseteq T$, $L_{f(S)} = L$. A *special* l.u.b. function is one that reserves a set for every L in F. (Clearly, if f is special, then f is full-range.)

Proposition 3. If f is a computable special l.u.b. function for F, then the corresponding f-algorithm is an

RCC learner for F.

Proof. Let $L \in F$, and let T be reserved for L by f. Given a presentation $s_1, s_2, \ldots$ of L, let h be the smallest integer such that $T \subseteq \{s_1, s_2, \ldots, s_h\}$. Then, either $L_{g_h} = L$, in which case the f-algorithm has already reached the correct guess, or there will be an integer $k > h$ such that s_k is not in L_{g_h}. Then, $g_k = f(\{s_1, \ldots, s_k\})$ will be such that $L_{g_k} = L$, since $T \subseteq \{s_1, \ldots, s_k\}$ and f is special. □

Although not every computable full-range l.u.b. function f gives rise to a learning algorithm, from every such f one can effectively obtain a special l.u.b. function $\bar{f}$ and, from Proposition 3, a learning $\bar{f}$-algorithm.

Proposition 4. Let f be a computable full-range l.u.b. function for the family F. Then, there is a special l.u.b. function $\bar{f}$ for F, which can be computed using the procedure for f as a subroutine.

Proof. Let $W_1, W_2, W_3, \ldots$ be a recursive enumeration of all finite subsets of Σ^*. Let us consider the set $\{j : W_j \subseteq S \text{ and } S \subseteq L_{f(W_j)}\}$. If $S \subseteq L$ for some L in F, then this set is not empty since it contains at least that j such that $W_j = S$. It is also clear that the minimum of this set, $m = \min\{j : W_j \subseteq S \text{ and } S \subseteq L_{f(W_j)}\}$, is computable and so is the function $\bar{f}$ defined as $\bar{f}(S) = f(W_m)$.

Clearly, $\bar{f}$ is an l.u.b. function. We now show that $\bar{f}$ is special. Let $k = \min\{i : L_{f(W_i)} = L\}$. (Such a k exists since f is full-range.) We observe that, if $j < k$ and if $W_j \subseteq L$, then $L_{f(W_j)} \not\supseteq L$. In fact, $L \neq L_{f(W_j)}$ by the definition of k, since $j < k$. Moreover, $L_{f(W_j)} \not\supset L$ otherwise $L_{f(W_j)}$ would not be an l.u.b. of W_j. Thus, for $j < k$ and $W_j \subseteq L$, there is a $x_j \in (L - L_{f(W_j)})$. Let V be a finite set containing a string $x_j \in (L - L_{f(W_j)})$, for each $j < k$ such that $W_j \subseteq L$. We claim that the set $T = W_k \cup V$ is reserved by $\bar{f}$ for L. In fact, if $T \subseteq S \subseteq L$, then for $j < k$ either $W_j \not\subseteq S$ or $S \not\subseteq L_{f(W_j)}$. On the other hand, $W_k \subseteq S \subseteq L_{f(W_k)} = L$ so that, by the definition of $\bar{f}$, $\bar{f}(S) = f(W_k)$. □

Given a computable full-range l.u.b. function f, Propositions 4 and 3 allow us to construct an RCC learner. The reverse process is considered in the next proposition.

Proposition 5. Given an RCC learner for a family F, one can construct a full-range l.u.b. function for F.

Proof. Let S be a subset of some language in F, and let $g_{|S|}$ be the output of an RCC algorithm whose input $s_1, s_2, \ldots, s_{|S|}$ is the lexicographic enumeration of S. Clearly $g_{|S|} \neq 0$, since the algorithm is responsive. Moreover, for no L in F we can have $S \subseteq L \subset L_{g_{|S|}}$, or otherwise the algorithm would not learn L, due to conservativeness. Thus, $f(S) \triangleq g_{|S|}$ is an l.u.b. function. Further, f is a full-range function because the entire family is learned by this algorithm and every language appears as a guess when it is presented. (Actually, in addition f turns out to be special.) □

Propositions 3, 4 and 5, imply the following characterization of RCC learnable families.

Theorem 2. There exists a responsive and consistent conservative learning algorithm for an indexed family F if and only if there is a computable full-range l.u.b. function for F.

4 RCC Learning of Special Families

In this section we consider RCC learnable families with specific properties which give rise to interesting consequences. Angluin (1980) considered families in which every finite set is contained in at most a finite number of languages, and showed that any f-algorithm is a learner for such families. We generalize this result by introducing the following notion, which refines that of the tell-tale subset.

Definition 3. A *private tell-tale* subset T of a language L in a family F is a finite subset of L such that, if $T \subseteq S \subseteq L$, then S is not a tell-tale subset of any language in F other than L. (In any family with *finite elasticity* (Wright, 1989), every language has private tell-tale subsets.)

It is easy to see that if all languages of F have private tell-tale subsets, then any l.u.b. function for F must be special. Hence, we obtain the following result.

Proposition 6. If every language in F has a private tell-tale subset, then the f-algorithm corresponding to any l.u.b. function f for F is a learner.

Another interesting class is that of the families that can be effectively enumerated in topologically sorted order. A *topologically sorted enumeration* of a family F is an enumeration of indices (at least one for each language in F) with the property that if L_i is a strict subset of L_j, then i appears before j in the enumeration. There is a topologically sorted enumeration of F if and only if no language of F includes infinitely many other languages of F.

Proposition 7. If an indexed family F admits a topologically sorted recursive enumeration, then it is RCC learnable.

Proof. Let f be the function that associates to each finite set S contained in some $L \in F$, the first index i in the topologically sorted recursive enumeration of F such that $S \subseteq L_i$. Clearly, $f(S)$ is computable from S, and it is a simple exercise to show that f is a special l.u.b. function for F. The conclusion then follows from Proposition 3. □

We observe that there are families, such as $F = \{L_k : k \geq 1, L_k = \{k, k+1, \ldots\}\}$, which are RCC learnable, but have no topological sorting.

A family of languages is *linearly ordered* if, for any two languages in it, one is a subset of the other. Linearly ordered families may have special significance for language acquisition in the parameter-view of natural language. (Cf. Manzini and Wexler (1987).) In a linearly ordered family, the l.u.b. of a set, if it exists, is unique. Moreover, it is the l.u.b. of one of the strings in the set.

Proposition 8. If $s_k \notin l.u.b.\{s_1, \ldots, s_{k-1}\}$, then $l.u.b.\{s_1, \ldots, s_k\} = l.u.b.\{s_k\}$.

Proof. Clearly, $l.u.b.\{s_1, \ldots, s_{k-1}\} \subset l.u.b.\{s_k\}$, for if the inclusion were reversed s_k would belong to $l.u.b.\{s_1, \ldots, s_{k-1}\}$, against the hypothesis. Thus, $\{s_1, \ldots, s_k\} \subseteq l.u.b.\{s_k\}$, and $l.u.b.\{s_1, \ldots, s_k\} \subseteq l.u.b.\{s_k\}$. Since a larger set cannot have a strictly smaller l.u.b., equality follows. □

Proposition 8 leads to a characterization for RCC learnability of linearly ordered families based only on the l.u.b. languages of singleton sets.

Proposition 9. A linearly ordered indexed family F is RCC learnable if and only if there exists a recursive function ϕ with the following properties:

(i) ϕ is defined for every string x which belongs to some L in F;

(ii) $L_{\phi(x)}$ is an l.u.b. for the set $\{x\}$;

(iii) For every L in F there is an x such that $L_{\phi(x)} = L$.

If ϕ satisfies the above property, an RCC learner is given by the algorithm A_ϕ that, at the ith stage, maintains its guess if consistent with the input string s_i, and changes it to $\phi(s_i)$ otherwise.

Proof. We first assume that there is an RCC learner for F and show that the function ϕ such that $\phi(x)$ is the index guessed by the learner when the first string presented to it is x satisfies the desired properties. Properties (i) and (ii) are trivially shown. for (iii), we argue as follows. Let L be an arbitrary language of F, and $s_1, s_2, \ldots$ a presentation of it. Let k be the smallest integer such that $L_{g_k} = L$. Since $g_{k-1} \neq g_k$, it must be the case that $s_k \notin l.u.b.\{s_1, \ldots, s_{k-1}\}$. Then, by Proposition 8, $L_{g_k} = l.u.b.\{s_1, \ldots, s_k\} = l.u.b.\{s_k\}$. In conclusion, $L = L_{\phi(x)}$ for some x (in this case $x = s_k$).

We now assume that a function ϕ with the properties (i), (ii), and (iii) exists, and show that the corresponding algorithm A_ϕ learns any language L in F. Indeed, by Proposition 8, at any stage i, the language L_{g_i} guessed by the algorithm is an l.u.b. of $\{s_1, \ldots, s_i\}$, and therefore $L_{g_i} \subseteq L$. By property (iii), at some stage k a string s_k such that $\phi(s_k) = L$ must appear in the input, and the guess converge to a correct value. $\square$

It is interesting to observe that algorithm A_ϕ in the above proposition need not store the input strings after having processed them. (The state of the algorithm is the current guess.)

5 Relation between Variants of Conservative Learning

Dropping the consistency requirement opens, for the learning algorithm, the possibility to guess a language inconsistent with the input. However, if a consistent guess is made, then it must be an l.u.b. of the input, or the conservative constraint will forbid the algorithm to recover from a guess which is actually a superset of the right language. Similarly, dropping the responsiveness requirement permits the learner not to guess after every input string. However, if a guess is made, then it must be an l.u.b. of the input.

In the absence of the conservativeness constraint, Angluin (1980) showed that if F is learnable, it is also learnable by a consistent and responsive algorithm. In contrast, we next establish the nested relationship between the classes of families learnable by the various types of conservative learners.

Theorem 3. (1) If there is a conservative learning algorithm for F, then there also is a consistent conservative learning algorithm for F.

(2) There is a family F for which there exists a consistent, conservative learning algorithm, but no responsive, conservative learning algorithm.

(3) There is a family F for which there exists a responsive, conservative learning algorithm, but no consistent and responsive conservative learning algorithm.

Proof. (1) We modify the conservative learner to refrain from making a guess whenever it would make an inconsistent guess. It is easy to see that the modified learner learns the original family and in addition is consistent.

(2) Consider the family F: for all $k \geq 1, L_k = \{0\} \cup \{k, k+1, ...\}$. This family has a consistent, conservative learner for which $g_n = \min(S_n - \{0\})$ with $\min(\emptyset)$ defined as 0. However, F does not have a responsive, conservative learner because on the input $\{0\}$ none of the indices can be guessed safely.

(3) Consider the family F: for all $k > 1$, $L_k = \{0\} \cup \{k, k+1, ...\}$, and $L_1 = \{5\}$. By the argument given in the proof of part (2) of this theorem, there is no consistent, responsive, conservative learning algorithm for F. We do have a responsive, conservative learner which behaves as follows: If $s_1 = s_2 = \cdots = s_n = 0$ or $s_1 = s_2 = \cdots = s_n = 5$, then $g_n = 1$, else $g_n = \min(S_n - \{0\})$. □

In the next section, we consider in detail the situation in which the conservative learner is constrained to be responsive but not necessarily consistent. Consistent and conservative learning will be investigated in Section 7.

6 Responsive and Conservative Learning

At the outset we have good reason to expect that the condition characterizing the responsive and conservative case would be the existence of some weak form of a computable full-range l.u.b. function. If we try to extract such a function from an inconsistent learner, we straightaway observe that we would be forced to weaken the function to take inconsistent values, i.e. where the domain set is not a subset of the function value. Furthermore, if there is a finite set S which happens to be a language in F, the lexicographic presentation of S need not force the learner to guess an index for the language S. (See Proposition 10 below for details required to adjust to this difference.) In view of this we introduce a parameter k, further weakening the l.u.b. function, but thereby making it computable from a responsive and conservative learner. (Note that this is similar to the situation in case of general *set-driven* learning. While any family with only infinite languages is learnable in a set-driven fashion, a general family may be learnable only in a *partially set-driven* fashion. (For more on these notions, see Osherson, Stob & Weinstein, 1986.) Our results in this section could also be simplified for the special case of only infinite languages in the family.)

Definition 4. Let $g(S,k)$ be a positive integer function defined whenever S is a finite subset of a language in F and for any positive integer k. We say that g is a *pseudo multi-l.u.b.* function for F if, for any k and S, no proper subset of $L_{g(S,k)}$ containing S belongs to F. (The language $L_{g(S,k)}$ is not constrained to contain S, but if it does then it must be an l.u.b. of S.) A pseudo multi-l.u.b. function is *full-range* if each L in F has a subset S such that, for some k, $L_{g(S,k)} = L$.

Proposition 10. Given a responsive and conservative learner for a family F, one can construct a full-range pseudo multi-l.u.b. function for F.

Proof. Define $g(S,k)$ to be the output of the learner whose input is the lexicographic enumeration $s_1, s_2, \ldots, s_n$ of S ($n = |S|$) followed by $k-1$ repetitions of s_n. Given the conservative nature of the learner, either $S \not\subseteq L_{g(S,k)}$ or $L_{g(S,k)}$ is an l.u.b. of S. Hence, g is a pseudo multi-l.u.b. function. To establish the full-range property we argue as follows.

If $L \in F$ is infinite, let $s_1, s_2, s_3, \ldots$ be the lexicographic presentation of L. For some i, the learner must output an index of L. Therefore $L_{g(\{s_1, \ldots, s_i\}, 1)} = L$. If $L \in F$ is finite, let $s_1, s_2, \ldots, s_n$ be the lexicographic enumeration of L, and let $s_1, s_2, \ldots, s_n, s_n, s_n, \ldots$ be a presentation of L. For some i, the learner must output an index for L. If $i \leq n$, then $L_{g(\{s_1, \ldots, s_i\}, 1)} = L$, else $(i > n)$, $L_{g(\{s_1, \ldots, s_n\}, i-n+1)} = L$. □

Propositions 11 and 12 below, which are analogous to Propositions 3 and 4, respectively for RCC learners, allow us to construct a responsive and conservative learner given a computable full-range pseudo multi-l.u.b. function. We begin with two definitions.

Definition 5. Let g be a pseudo multi-l.u.b. function for family F. Let $L \in F$, and T be a finite subset of L. We say that T is *reserved* for L (beyond k) if, for every finite $S \supseteq T$ and every $m \geq k$, $L_{g(S,m)} = L$. A *special* pseudo multi-l.u.b. function is one that reserves a set for every L in F. (Clearly, if g is special, then g is full-range.)

Definition 6. Given a computable multi-l.u.b. function g for an indexed family F, the guessing algorithm that, at the ith stage, maintains its guess if consistent with the previous input set as well as s_i, and changes it to $g(\{s_1, s_2, \ldots, s_i\}, i)$ otherwise, is called a g-algorithm.

Proposition 11. If g is a computable special pseudo multi-l.u.b. function for F, then the corresponding g-algorithm is a responsive and conservative learner for F.

Proof. Let $L \in F$, and let T be reserved (beyond k) for L. Given a presentation $s_1, s_2, \ldots$ of L, let h be the smallest integer such that $h \geq k$ and $T \subseteq \{s_1, s_2, \ldots, s_k\}$. Then, either $L_{g_h} = L$, in which case the g-algorithm has already reached the correct guess, or there will be an integer $m > k$ such that s_m is not in L_{g_h}. Then $g_k = g(\{s_1, \ldots, s_m\}, m)$ will be an index for L because T was reserved by g beyond k and thus L will indeed get learned. □

Proposition 12. Let g be a computable full-range pseudo multi-l.u.b. function for the family F. Then there is a special pseudo multi-l.u.b. function $\bar{g}$ for F, which can be computed using the procedure for g as a subroutine.

Proof sketch. Consider a recursive enumeration $X_1, X_2, \ldots$ of the pairs (S,k)s where S is a finite subset of Σ^* and k a positive integer. Let $X_j = (W_j, h_j)$. Let us define $\bar{g}(S,k)$ as follows. We consider the set $\{j : j \le k, W_j \subseteq S \text{ and } S \subseteq L_{g(W_j,h_j)}\}$. If this set is not empty we let i be its minimum element, and we let $\bar{g}$ equal $g(W_i,h_i)$. Otherwise we let $\bar{g}$ be $g(S,k)$. It can be argued that $\bar{g}$ as defined here is a special pseudo multi-l.u.b. function. □

By combining the above propositions we obtain the following characterization for responsive conservative learning.

Theorem 4. There exists a responsive conservative learning algorithm for an indexed family F if and only if there is a computable full-range pseudo multi-l.u.b. function for F.

7 Consistent and Conservative Learning

In this section, we give two characterizations of families learnable by conservative (and consistent, due to Theorem 3 (1)) learners. The first characterization, in terms of functions with suitable l.u.b. properties, is parallel to that of Section 5 and is reported without proofs. The second characterization is in terms of tell-tale subsets and is developed in greater detail.

Definition 7. Let g be a nonnegative integer function defined on all finite subsets of the languages in F and positive integers. We say that g is an *incomplete multi-l.u.b.* function for F if, for all integers k and finite sets S, either $g(S,k)$ is 0 or $L_{g(S,k)}$ is a least upper bound of S. An incomplete multi-l.u.b. function is *full-range* if each L in F has a subset S such that, for some k, $L_{g(S,k)} = L$.

Theorem 5. There exists a conservative and consistent learning algorithm for an indexed family F if and only if there is a computable full-range incomplete multi-l.u.b. function for F.

For conservative and consistent learning another characterization can be developed, in the spirit of the Angluin result reported in Theorem 1 for general learning. In essence, we show that conservative learning is equivalent to the possibility of *computing* tell-tale subsets, rather than just recursively enumerating them. However, when comparing Angluin's result on general learning (Theorem 1) with our result on conservative learning (Theorem 6, below), the following difference should be observed. In both cases, it is assumed that the procedure that produces the tell-tale subsets can use a given learner as a subroutine. For general learning, the tell-tale subset of a language L learned by the learner can be obtained from any total machine that accepts L. For conservative learning instead, a tell-tale subset for L can be obtained only if L is specified by one of the indices for L that the learner actually outputs for at least some input sequence.

Proposition 13. Given a conservative consistent learner for an indexed family F, it is possible to construct a procedure A that, on input i (an index) will behave as follows:

(a) If the learner ever outputs index i as a guess corresponding to some input sequence, then A outputs a tell-tale subset of L_i and then halts;

(b) otherwise, A does not halt and does not produce any output.

Proof. Let $Z_1, Z_2, \ldots$ be a recursive enumeration of all finite sequences of strings in Σ^* (repetitions allowed). On input i, Procedure A enumerates $Z_1, Z_2, \ldots$ and, for a given j, it checks whether each string of Z_j is in L_i and whether the conservative learner outputs i on input Z_j. If so, A outputs (the set underlying) Z_j and then halts. Properties (a) and (b) are easily established. □

In the following definition, we isolate properties of procedure A in Proposition 13 that can be stated without reference to a particular learner, but are sufficient to guarantee conservative learnability.

Definition 8. A procedure A is a *full-range tell-tale set enumerator* for an indexed family F if (a) whenever A halts on input i, the output of A is a tell-tale set for L_i, and (b) for each L in F, A halts on at least one index i such that $L_i = L$.

Theorem 6. There exists a conservative and consistent learning algorithm for an indexed family F if and only if there is a full-range tell-tale set enumerator for it.

Proof. That a full-range tell-tale set enumerator can be obtained from a conservative and consistent learning algorithm follows from Proposition 13.

Let A be the full-range tell-tale set enumerating algorithm. The learning algorithm is as follows (where for convenience we assume the notation $g_0 = 0$, and $L_0 = \emptyset$):

Stage n ($n \geq 1$) {Let $S_n = \{s_1, s_2, \ldots, s_n\}$}

```
if S_n ⊆ L_{g_{n-1}} then g_n := g_{n-1}
else
    begin
    m:=1;
    g_n:=0;
    while (m ≤ n) and (g_n = 0) do
       begin
       if A has halted on m in n steps producing T_m then
          if (T_m ⊆ S_n) and (S_n ⊆ L_m) then
             g_n := m;
       m:=m+1
       end
    end;
Go to stage n+1.
```

It is easy to verify that the above algorithm is conservative and consistent. We next show that it is correct. Let $i \geq 1$ be arbitrary and let $\sigma = s_1, s_2, s_3, \ldots$ be a positive presentation of L_i. At any stage n, if $g_n \neq 0$, then S_n is a tell-tale subset of L_{g_n} and therefore L_i, which contains S_n, can not be a subset of L_{g_n}. Thus, if $L_{g_n} \neq L_i$, at some stage after the nth one, the input string will not belong to L_{g_n}, forcing a change of guess (the *else* clause is executed). Let k be the smallest index for L_i such that algorithm A halts on input k after outputting T_k (a tell-tale subset for L_i). Consider the state of the algorithm subsequent to the stage at which all of T_k (1) has been outputted by A on k, and (2) has appeared in σ. Since k meets the requirements of the **else** portion, the value of m in the **while** loop will not exceed k. Thus there are only a bounded number of possible guesses left for the learner, and k has to be guessed, unless an index for L_i

has already been guessed. □

Acknowledgment. The authors wish to thank Kieran Herley for his comments on a previous draft of this paper.

References

ANGLUIN, D. (1980) Inductive inference of formal languages from positive data. *Information and Control*, 48, 117-135.

ANGLUIN, D & SMITH, C. H. (1983) Inductive inference: theory and methods. *Computing Surveys*, 15(3):237-269, September 1983.

BERWICK, R. (1985) *The Acquisition of Syntactic Knowledge*. MIT press, Cambridge .

CHOMSKY, N. (1986) *Knowledge of Language*. Praeger.

GOLD, E. M. (1967) Language identification in the limit. *Information and Control*, 10, 447-474.

KAPUR, S., LUST, B., HARBERT, W. & MARTOHARDJONO, G. (1989) On relating Universal Grammar and Learnability theory: the case of Binding Domains and the Subset Principle. *Proceedings of the Conference on Knowledge and Language*, Groningen, The Netherlands.

LUST, B. (1987) Remarks on the psychological reality of the subset principle: Its relation to universal grammar as a model of the initial state. in *Proceedings of the 1986 Cognitive Science Meeting*, Lawrence Rlbaum Associates, Hillsdale, New Jersey, 420-431.

MANZINI, R. M. & WEXLER, K. (1987) Parameters, Binding Theory and Learnability. *Linguistic Inquiry*, 18, 3, 413-444, Summer 1987.

OSHERSON, D. N., STOB, M. & WEINSTEIN, S. (1986) *Systems that Learn: An Introduction to Learning*. MIT press, Cambridge.

SOARE, R. (1987) *Recursively Enumerable Sets and Degrees*. Springer-Verlag.

WILLIAMS, E. (1987) Introduction in *Parameter Setting*, Eds. Thomas Roeper and Edwin Williams, D. Reidel Publishing Co.

WRIGHT, K. (1989) Identification of unions of languages drawn from an identifiable class. *Proceedings of the Second Conference on Computational Learning Theory*, Santa Cruz, California. Morgan Kaufmann publishers.